REMARKS/ARGUMENTS

This preliminary amendment is in response to the Final Office Action dated August 22, 2005. Claims 1-12 are pending in the present application. Claims 1-12 have been rejected. Claim 3 has been canceled. Claims 1, 2, and 4-12 remain pending. Claim 1 has been amended to incorporate the limitations of dependent claim 3. Applicants respectfully submit that no new matter has been presented. For the reasons set forth more fully below, Applicants respectfully submit that the claims as presented are allowable. Consequently, reconsideration, allowance, and passage to issue are respectfully requested.

In the event, however, that the Examiner is not persuaded by Applicants' amendments and arguments, Applicants respectfully request that the Examiner enter the arguments to clarify issues upon appeal.

Claim Rejections - 35 U.S.C. §103

The Examiner has stated:

> **Claims 1-12 are rejected under 35 U.S.C. 103(a) as being unpatentable over Duxbury et al. (hereafter Duxbury) (U.S. Patent 5604896), in view of Baisley et al. (hereafter Baisley) (U.S. Patent 6292932 B1).**
>
> **As to claim 1, Duxbury teaches the invention substantially as claimed including a method of processing an application request on an end user application and an application server including a mapping support language comprising the steps of:**
>
> **a. Initiating the application request on the end user application in a first language with a first application program [col. 1, lines 32-34];**
> **b. Transmitting the application request back to the server and converting the application request from the first language of the first end user application to a form for the mapping support language running on the application server [col. 1, lines 35-37];**
> **c. Processing said application request on the application server [col. 3, lines 15-18];**
> **d. Transmitting a response to the application request from the application server to the end user application, and converting the response to the application request from the mapping support language running on the**

application server to the first language of the first end user application [col. 3, lines 23-27];

e. Wherein the end user application and the application server have at least one connector therebetween [col. 2, line 36], and the steps of (i) converting the application request from the first language of the first end user application as a source language to the language running on the application server as a target language, and (ii) converting a response to the application request from the language running on the application server as a source language to the first language of the first end user application as a target language [col. 8, lines 45-50].

Duxbury does specifically not teach:

a. invoking connector metamodels of respective source language and target mapping support language;
b. populating the connector metamodels with metamodel data of each of the respective source language and target mapping support language, the metamodel data of the target mapping support language including a map, mapset, and a mapfield; and
c. converting the source language to the mapping support language.

However, Baisley teaches:

a. A source metamodel [col. 3, line 39] and a target mapping support language [col. 3, 40-41].
b. the UML model has other information, either as tags or in an external representation [col. 3, lines 48-50] and the metamodel maps data type and attributes in derived data types, maps the derived attributes to aliases of the supertype [col. 4, lines 3-8].
c. converts UML model to a MOF model [col. 3, line 44].

It would have been obvious to one of ordinary skill in the art at the time the invention was made to have combined the teaching of Duxbury and Baisley because Baisley's method of using metamodel for the mapping support would provide a predictable and reliable mapping between the original source language and the target language to Duxbury's system.

As to claim 2, Duxbury teaches the end user application is a web browser [col. 2, line16]...

Response to Arguments

Applicant's arguments filed 03/24/2005 for claims 1-12 have been fully considered but are not persuasive.

In the remarks, Applicant argued in substance that:

a. There is no mention of "converting a response to the application request from the language running on the application request from the language running on the application server as a source language to the first language of the first end user application as a target language".

b. Duxbury does not teach 1) invoking connector metamodels of respective source language and target mapping support language; 2) populating the connector metamodels with metamodel data of each of the respective source language and target mapping support language, the metamodel data of the target mapping support language including a map, mapset, and a mapfield; and 3) converting the source language to the mapping support language.

Examiner respectfully traversed Applicant's remarks:

c. As to point (a), Dialogue is a conversation between two people, and language is a tool for conversation. Therefore, convert a dialogue constitutes convert a language. Duxbury teaches "converts this request into a dialogue

("language") with the remote ("target") application, so as to perform the desired transfer and to send a response back to the client". Therefore, Duxbury clearly teaches "converting" from the source language to the target language.

d. As to point (b), 1) the invoking of the connector of the metamodel is obvious since, based on the succeeding steps, if the connector of the metamodel was not invoked, there was no metamodel to be populated 2) Baisley discloses populating the metamodels with metamodel data as states in col. 3, lines 47-49 "any other information needed can be recorded ("populating") either as tags in the UML model or in an external representation". 3) In addition, Baisley discloses mapfields ("elements of the UML model") are mappled to instances ("mapset") of MOF Data Type [col. 3, lines 55-57], and map ("alias") [col. 4, lines 7-16]...

Applicants respectfully disagree with the Examiner's rejections. The present invention provides a method of processing an application request on an end user application and an application server including a mapping support language. In accordance with the present invention, the method includes: a) initiating the application request on the end user application in a first language with a first application program; b) transmitting the application request to the server and converting the application request from the first language of the first end user application to a form for the mapping support language running on the application server, wherein the end user application is connected to the application server through a web server, and the web server comprises a connector; c) processing said application request on the application server; and d) transmitting a response to the application request from the application server to the end user application, and converting the response to the application request from the mapping support language running on the application server to the first language of the first end user application. The end user application and the application server have at least one connector therebetween, and the steps of: (i) converting the application request from the first language of the first end user application as a source language to the language running on the application server as a target language, and (ii) converting a response to the application request from the language running on the application server as a source language to the first language of the first

end user application as a target language. Each includes the steps of: 1) invoking connector metamodels of respective source language and target mapping support language; 2) populating the connector metamodels with metamodel data of each of the respective source language and target mapping support language, the metamodel data of the target mapping support language including a map, a mapset, and a mapfield; and 3) converting the source language to the mapping support language. Duxbury in view of Baisley does not teach or suggest these features, as discussed below.

Duxbury discloses a computer with a terminal emulation interface for multi-environment client/server applications. A computer system includes first and second processing environments interconnected by a gateway. The gateway emulates a terminal in the second environment, converting service requests from a client in the first environment into dialogues on the emulated terminal by executing scripts in a scripting language. This allows client applications in the first environment to communicate with server applications in the second environment in a way that is completely transparent to the clients. The client is not aware that it is communicating with the server through a dialogue on an emulated terminal; all knowledge of the dialogue is embodied in the scripts. This is of utility in integrating legacy computer systems with new systems. (Abstract).

Applicants respectfully submit that Duxbury does not teach or suggest "converting the application request from the first language of the first end user application to a form for the mapping support language running on the application server," as recited in independent claims 1, 5, and 8. The Examiner has asserted that:

> dialogue is a conversation between two people, and language is a tool for conversation. Therefore, convert a dialogue constitutes convert a language.

> Duxbury teaches "converts this request into a dialogue ("language") with the remote ("target") application, so as to perform the desired transfer and to send a response back to the client". Therefore, Duxbury clearly teaches "converting" from the source language to the target language.

However, applicants respectfully assert that Duxbury does not teach a first language of the first end user application (e.g., Java or C or C++) to a form for the mapping support language running on the application server (e.g., COBAL). Instead, Duxbury is merely executing scripts in a scripting language. While Duxbury describes a dialog that occurs in a script (Abstract), nowhere does Duxbury describe that there is any translation from one language to another language in the script. Duxbury mentions converting parameters of a request into shell variables. However, converting from one form to another form (i.e., parameters to variables) is clearly not the same as converting from one language to another language (i.e., C++ to COBAL). In contrast to the present invention, Duxbury does not teach or such suggest how to handle different requests in different languages (e.g., Java and C and C++).

Applicants agree with the Examiner that Duxbury does not teach 1) invoking connector metamodels of respective source language and target mapping support language; 2) populating the connector metamodels with metamodel data of each of the respective source language and target mapping support language, the metamodel data of the target mapping support language including a map, mapset, and a mapfield; and 3) converting the source language to the mapping support language.

The Examiner has referred to Baisley as teaching the steps that are not described in Duxbury. Baisley discloses a system and method for converting from one modeling language to another. A method is disclosed for converting a UML model to a MOF model within a repository in a computing system having a repository program being executed by said system and

a means for storing data. The method includes the steps of selecting a package within the UML model (or "UML package") to be exported to the MOF model. Next, the UML package and its elements are exported to the MOF model. Next, relations are recursively set between MOF objects of the UML package that correspond to relations between UML objects in the package. Next, MOF reference objects are created for navigable MOF association ends. (Abstract).

However, Applicants respectfully submit that Baisley fails to teach or suggest the combination of the "invoking" and "populating" elements as claimed. Specifically, Baisley fails to teach or suggest, "invoking connector metamodels of respective source language and target mapping support language," as recited in independent claims 1, 5, and 8. The Examiner has asserted that "if the connector of the metamodel was not invoked, there was no metamodel to be populated." However, Applicants fail to understand how this hypothetical is evidence that Baisley teaches the invoking step. The Examiner has referred to a source metamodel (column 3, lines 39) and a target mapping support language (column 3, lines 40-41) of Baisley. However, Baisley is merely describing modeling languages that are "used to model metamodels." Baisley does not specifically describe invoking metamodels of "respective source language and target mapping support language," as claimed.

Furthermore, Baisley fails to teach or suggest, "populating the connector metamodels with metamodel data of each of the respective source language and target mapping support language, the metamodel data of the target mapping support language including a map, a mapset, and a mapfield," as recited in independent claims 1, 5, and 8. The Examiner has referred to column 3, lines 47-49 of Baisley, which states that "any other information needed can be recorded, either as tags in the UML model or in an external representation." However, Baisley is

specifically discussing recording information in a single model (i.e., a UML model). Baisley is clearly not discussing metamodels with metamodel data of multiple source languages and a target mapping support language as in the present invention.

With regard to "the metamodel data of the target mapping support language including a map, a mapset, and a mapfield," the Examiner has referred to "elements of the UML model" and "instances" of Baisley as teaching the "mapfields" and "mapset" of the present invention, respectively (column 3, lines 55-57), and has referred to an "alias" of Baisley as teaching the "map" of the present invention (column 4, lines 7-16). However, Baisley merely states in column 3, lines 55-57 that "Elements of the UML model that are intended to represent data types are mapped to instances of MOF Data Type, using the following rules...." The MOF data types of Baisley are specifically associated with the MOF model, which is merely a modeling language used to model metamodels. In contrast to the data types of Baisley, the target mapping support language as claimed is associated with an application request.

The Examiner has also referred to Baisley, column 3, lines 48-50, and column 4, lines 3-8, as teaching this element. However, nowhere does Baisley disclose the elements of "metamodel data of the target mapping support language, including a map, a mapset, and a mapfield" as claimed. Instead, Baisley states:

> 4. For data types with attributes, map to a MOF DataType representing a corresponding structure.
>
> 5. However, for data types that contain a single attribute, which does not denote significant meaning beyond the data type itself, the attribute may be ignored. In this case, the type of the attribute is used to determine a direct mapping from the UML data type to the a MOF data type, representing an alias of the mapped type.

As shown, Baisley fails to specifically describe the "target mapping support language" that includes "map, a mapset, and a mapfield," as claimed. The data types as disclosed in Baisley are not the same as the "target mapping support language" that includes the "map, a mapset, and a mapfield." As described above, the data types of Baisley are specifically associated with the MOF model, which is merely a modeling language used to model metamodels. In contrast, the target mapping support language as claimed is associated with an application request.

Therefore, Duxbury in view of Baisley does not teach or suggest each and every element as recited in independent claims 1, 5, and 8, and these claims are allowable over the cited references.

Independent claim 11

Similar to independent claims 1, 5, and 8, independent claim 11 recites a "target mapping support language" that includes "map, a mapset, and a mapfield." As described above, with respect to independent claims 1, 5, and 8, Duxbury in view of Baisley does not teach or suggest these elements. Accordingly, the above-articulated arguments related to independent claims 1, 5, and 8 apply with equal force to claim 11. Therefore, claim 11 is allowable over Duxbury in view of Baisley for at least the same reasons as claims 1, 5, and 8.

Dependent claims

Dependent claims 2, 4, 6-7, 9-10, and 12 depend from independent claims 1, 5, 8, and 11, respectively. Accordingly, the above-articulated arguments related to independent claims 1, 5,

8, and 11 apply with equal force to claims 2, 4, 6-7, 9-10, and 12, which are thus allowable over the cited references for at least the same reasons as claims 1, 5, 8, and 11.

Conclusion

In view of the foregoing, Applicants submit that claims 1, 2, and 4-12 are patentable over the cited references. Applicants, therefore, respectfully request reconsideration and allowance of the claims as now presented.

Applicants' attorney believes that this application is in condition for allowance. Should any unresolved issues remain, the Examiner is invited to call Applicants' attorney at the telephone number indicated below.

Respectfully submitted,

SAWYER LAW GROUP LLP

December 12, 2005
Date

Joseph A. Sawyer, Jr.
Attorney for Applicant(s)
Reg. No. 30,801
(650) 493-4540